1.代码题

回溯法求子集

class Solution {

public:

vector<vector<int>> res;

void backtrack(vector<int> &track, int start, vector<int>& nums) {

res.push\_back(track);

for(int i = 0; i < nums\_size; ++ i) {

track.push\_back(nums[i]);

backtrack(track,i+1,nums);

track.pop\_back();

}

}

vector<vector<int>> subsets(vector<int>& nums) {

vector<int> track;

backtrack(track,0,nums);

return res;

}

};

回溯法去重求子集

class Solution {

public:

vector<vector<int>> ans;

vector<int> cur;

vector<int> v;

void dfs(int idx) {

ans.push\_back(cur);

if(idx == v.size()) {

return;

}

for(int i=idx; i<v.size(); ++i) {

if(i > idx && v[i] == v[i-1]) continue;

cur.push\_back(v[i]);

dfs(i+1);

cur.pop\_back();

}

}

vector<vector<int>> subsetsWithDup(vector<int>& nums) {

sort(nums.begin(), nums.end());

v = nums;

dfs(0);

return ans;

}

};

2.知识点

分页和分段有什么区别（内存管理）？

　　段式存储管理是一种符合用户视角的内存分配管理方案。在段式存储管理中，将程序的地址空间划分为若干段（segment），如代码段，数据段，堆栈段；这样每个进程有一个二维地址空间，相互独立，互不干扰。段式管理的优点是：没有内碎片（因为段大小可变，改变段大小来消除内碎片）。但段换入换出时，会产生外碎片（比如4k的段换5k的段，会产生1k的外碎片）

　　页式存储管理方案是一种用户视角内存与物理内存相分离的内存分配管理方案。在页式存储管理中，将程序的逻辑地址划分为固定大小的页（page），而物理内存划分为同样大小的帧，程序加载时，可以将任意一页放入内存中任意一个帧，这些帧不必连续，从而实现了离散分离。页式存储管理的优点是：没有外碎片（因为页的大小固定），但会产生内碎片（一个页可能填充不满）。

两者的不同点：

目的不同：分页是由于系统管理的需要而不是用户的需要，它是信息的物理单位；分段的目的是为了能更好地满足用户的需要，它是信息的逻辑单位，它含有一组其意义相对完整的信息；

大小不同：页的大小固定且由系统决定，而段的长度却不固定，由其所完成的功能决定；

地址空间不同： 段向用户提供二维地址空间；页向用户提供的是一维地址空间；

信息共享：段是信息的逻辑单位，便于存储保护和信息的共享，页的保护和共享受到限制；

内存碎片：页式存储管理的优点是没有外碎片（因为页的大小固定），但会产生内碎片（一个页可能填充不满）；而段式管理的优点是没有内碎片（因为段大小可变，改变段大小来消除内碎片）。但段换入换出时，会产生外碎片（比如4k的段换5k的段，会产生1k的外碎片）。

3.开源项目

提交了一个pr在项目中，整理了项目文件夹